Hybrid approach for fast occlusion processing in computer generated hologram calculation

Antonin Gilles[1] Patrick Gioia[1,2] Rémi Cozot[1,3] Luce Morin[1,4]

[1] IRT b<>com Cesson-Sévigné France
[2] Orange Labs Rennes France
[3] University of Rennes 1 Rennes France
[4] INSA Rennes Rennes France

Abstract

A hybrid approach for fast occlusion processing in computer generated hologram calculation is studied in this paper. The proposed method is based on the combination of two commonly used approaches, which complement one another: the point-source and wave-field approaches. By using these two approaches together, the proposed method thus takes advantages from both of them. In this method, the 3D scene is first sliced into several depth layers parallel to the hologram plane. Light scattered by the scene is then propagated and shielded from one layer to another using either a point-source or a wave-field approach according to a threshold criterion on the number of points within the layer. Finally, the hologram is obtained by computing the propagation of light from the nearest layer to the hologram plane. Experimental results reveal that the proposed method does not produce any visible artifact and outperforms both the point-source and wave-field approaches.

Keywords: Computer-Generated Hologram, Color holography, Three-dimensional imaging

1 Introduction

Most current 3DTV systems are based on the stereoscopic technique. This technique provides three-dimensional illusion by delivering two different views of the scene to the respective left and right eyes of the viewer. The viewer thus perceives the three-dimensional effect by the binocular parallax activity. Thanks to its simplicity of implementation and its good compatibility with existing 2DTV systems, the stereoscopic technique quickly attracted a considerable attention [1]. However, this technology still presents many constraints and limitations such as the necessity of wearing special glasses for stereoscopic systems and the limited viewing zone of autostereoscopic systems [2]. Moreover, the stereoscopic technique does not provide monocular movement parallax and creates a mismatch between convergence and accommodation cues, which can lead to eye-strain and headaches [3].

To solve these limitations, several alternative technologies have been proposed in the last decades. Among these new techniques, holography is often considered as the most promising one, since it can provide realistic and natural three-dimensional illusion to the naked eye. Indeed, it provides all the human depth cues without the need for special viewing devices and without causing eye-strain [4].

However, a hologram is optically recorded by wave interference between two coherent laser beams in a dark room. The optical system must be kept very stable during hologram recording, since a very small vibration can destroy the interference fringes. Because of these requirements, conventional optical holography cannot be used for video and outdoor recording. To overcome optical hologram recording limitations, several methods have been proposed to generate holograms by computer calculation. Using these methods, it is possible to obtain Computer-Generated Holograms (CGH) of synthetic or existing scenes by simulating the propagation of light scattered by the scene towards the hologram plane. Depending on the method used, the resulting CGH may have various features.

One of the most important features in CGH techniques is the ability to properly take into account occlusions between objects in a scene. Indeed, occlusion is one of the most important cues in depth perception. Occlusion processing in CGH techniques, also called light shielding, is the counterpart of hidden surface removal in Computer Graphics (CG). However, unlike still CG images, CGH provides motion parallax, which is coupled with occlusion effect. Therefore, the occlusion effect in CGH implies that the visibility of objects changes according to the movement of the viewer.

Thanks to its attractive visualization properties, CGH may have application in the field of videoconferencing or telepresence systems. However, because of the huge calculation burden, real-time CGH computation is still very challenging. In this paper, we propose a new approach which aims at reducing the CGH calculation time to get closer to real-time computation. Two approaches are commonly used for CGH computation: the point-source and the wave-field approaches.

 http://dx.doi.org/10.1364/AO.55.005459

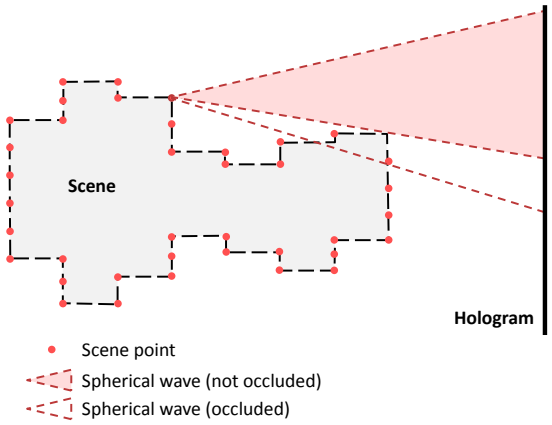

Figure 1: Light shielding using the point-source approach: a visibility test is done to check the existence of obstacles between an object point and each sampling point in the hologram plane.

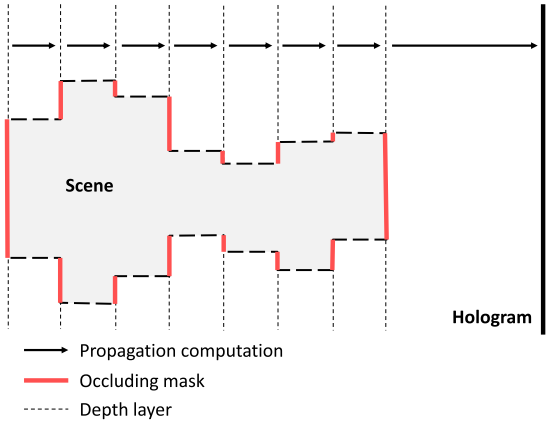

Figure 2: Light shielding using the wave-field approach based on a layered model of the scene: at each diffraction step, the light field is multiplied by the binary cross-section mask of the current depth layer.

1.1 Point-source approach

In the point-source approach [5], 3D scenes are sampled by a collection of self-luminous points and light propagation from the scene towards the hologram plane is computed as the sum of spherical waves scattered by each point. Light shielding is usually treated as a visibility test to check the existence of obstacles between an object point and each pixel of the hologram [6–9], as shown in Figure 1.

This approach is very flexible and does not impose any restriction on the scene geometry. However, its computational complexity is very high since it requires one calculation per point of the scene per pixel of the hologram. Moreover, to produce shapes that appear solid and continuous, the scene needs to be sampled at very high densities, making the CGH computation prohibitively slow. To reduce the computational complexity, several methods have been proposed [10–28].

In [10], the authors took advantage of the geometric symmetry of the spherical light wave equation to avoid redundant calculations. In this method, the complex wave scattered by each point is simultaneously calculated at four different pixels in the hologram plane, dividing the CGH calculation time by four. In [11] and [12], the authors proposed to compute the complex wave scattered by each point using recurrence formulas between adjacent pixels in the hologram plane rather than time-consuming direct calculation at each hologram pixel.

In [13, 14], the 3D scene is placed close to the hologram plane to reduce the number of hologram pixels enlightened by each scene point. In [15–18], the authors used the wavefront recording plane (WRP) method, in which a virtual plane is introduced between the 3D scene and the CGH. The complex wave scattered by each point is first calculated within the virtual plane. Since it is placed close to the 3D scene, each scene point illuminates only a few samples in the virtual plane. Finally, the complex wave in the virtual plane is propagated towards the hologram plane using diffraction formulas such as the Angular Spectrum propagation [29].

In [19], the author used a pre-computed look-up table (LUT) to store the wave scattered by scene points from each of the possible locations in the scene volume. Thus, the entire complex wave of a specific scene can be generated by fetching from the LUT the wave corresponding to each scene point and adding them together. In [20], the authors proposed a method to reduce the memory usage of the LUT. In this method, the 3D scene is sliced into a set of depth layers parallel to the hologram plane, and only the waves scattered by the center-located scene points on each layer are pre-calculated and stored in the LUT. Then, the waves scattered by other scene points on each layer are generated using shifting and scaling operations. In [21] and [22], the authors took advantage of the spatial redundancies within intensity and depth data of a 3D scene to reduce the number of scene points for which the complex wave has to be calculated.

In [23], the authors proposed to reduce the number of visibility tests needed for light shielding by grouping samples in the hologram plane. In [24, 25], the authors further reduced the number of visibility tests by grouping scene points. Finally, the CGH computation time can also be reduced using GPU hardware [26, 27] and special purpose hardware [28].

1.2 Wave-field approach

Another well-known CGH technique is the wave-field approach based on a layered model of the scene [30, 31]. In this approach, 3D scenes are sliced into depth layers parallel to the hologram. Light scattered by the scene is then propagated from one layer to another using diffraction formulas such as the Angular Spectrum propagation [29], as shown in Figure 2. Light shielding is performed in a similar way as the painter's algorithm in CG: at each diffraction step, the computed light field is multiplied by the binary cross-section mask of the current depth layer. This technique does not

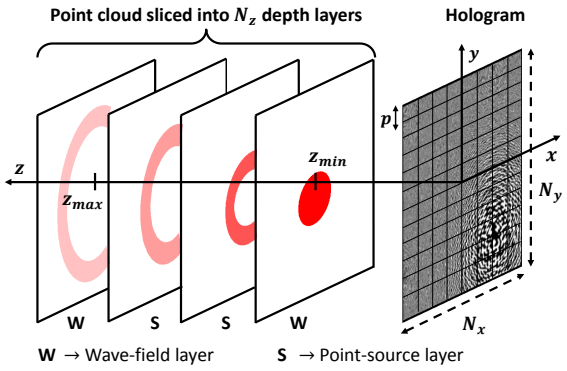

Figure 3: Scene geometry and coordinate system used by the proposed hybrid approach.

require any visibility test to be performed.

In [32], the authors generalized this approach to polygon-meshes. In this method, 3D scenes are described as a set of oriented polygons, each one being considered as a surface source of light. Light scattered by each polygon is numerically propagated using diffraction between tilted planes based on the Angular Spectrum of plane waves and coordinate rotation in the Fourier domain [33]. Occlusion processing is performed using the silhouette method: light scattered by the scene is shielded by each polygon using a binary silhouette mask function, which has value zero inside the orthogonal projection of the polygon on a plane parallel to the hologram plane and one elsewhere. Using this method, the authors successfully created full-parallax large-scaled CGHs [34].

The computation of the Angular Spectrum propagation involves the use of the Fast Fourier Transform (FFT) algorithm twice and is thus more time-consuming than the computation of the spherical light wave scattered by a single point. However, complex waves scattered by scene points located within a single planar segment are calculated all at once using the Angular Spectrum propagation. This approach is thus more efficient than the point-source approach when the number of scene points within each segment is sufficiently important. However, when the scene geometry contains complex shapes, a large number of polygons or depth layers containing only one or a few points are needed to sample it, making the wave-field approach less efficient than the point-source approach. To reduce the computational complexity of this approach, a technique that uses color-space conversion has been proposed [35].

1.3 Proposed hybrid approach

These two approaches complement one another: while the point-source approach is very efficient when the scene contains complex shapes, such as trees or human bodies, the wave-field approach is more efficient when objects in the scene consist of large planar surfaces, such as roads or buildings. However, most real 3D scenes contain both complex shapes and planar surfaces. As a consequence, these two approaches are rarely fully efficient for computing CGH of real scenes in their entirety.

To overcome these limitations, we propose a new CGH computation method with occlusion effect based on a fast hybrid point-source/wave-field approach. Whereas previously proposed methods aimed at reducing the computational complexity of the point-source or the wave-field approaches independently, the proposed method uses the two approaches together and therefore takes advantages from both of them.

A comparable method which uses multiple WRPs that are crossing through the scene has been proposed in [18]. In this method, the WRPs are processed sequentially. First, the complex waves scattered by scene points within a given WRP zone are computed and summed up in the corresponding WRP. Then, light is numerically propagated to the next WRP using Angular Spectrum propagation. This process is repeated until the last WRP has been reached, and light is finally propagated towards the hologram plane. This method thus uses an explicit point-source approach for intra-WRP light propagation and an implicit wave-field approach for inter-WRP light propagation. As such, it can be defined as a hybrid method. However, our algorithm differs from [18] with respect to two aspects. Firstly, in [18], the point-cloud is uniformly subdivided into a number of WRPs zones which is set depending on the depth extent of the scene and lateral density of the point-cloud. As a consequence, the WRPs may cross through only a few or none of the scene points. On the other hand, in our proposed method, the scene is sliced into several depth layers whose complex waves are computed using either a point-source or a wave-field approach depending on the number of scene points they contain, thus reducing the computational burden. Secondly, in [18] light shielding is performed by calculating the contribution of the points ordered from back to front. Before adding the contribution of a given point to a specific area of the WRP or the hologram plane, the optical field recorded in this area is suppressed by multiplying the field locally with the complement of a Gaussian distribution. This approximation is computationally very efficient, but it may produce artifacts for medium and large viewing angles. On the contrary, our proposed method enables exact occlusion of wave-field layers by point-source layers and vice-versa, at the cost of an increased computational complexity compared to [18].

The outlines of the method can be found in [36]. Compared to [36], this paper provides the mathematical demonstration of our algorithm, a detailed description of the GPU implementation and a thorough analysis of numerically and optically reconstructed scene images. Section 2 gives an overview of the proposed method, Section 3 and Section 4 present respectively light propagation and light shielding techniques used by the method, Section 5 explains how the depth layers are classified, and Section 6 gives a detailed description of the GPU implementation. Finally, the experimental results are analyzed in Section 7.

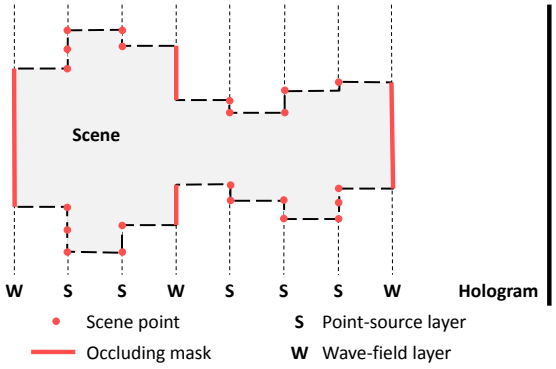

Figure 4: Hologram computation using the proposed hybrid approach.

2 Overview of the method

Figure 3 shows the scene geometry and coordinate system used by the proposed method. The coordinate system is defined by (x, y, z) so that the hologram lies on the $(x, y, 0)$ plane. The 3D scene is sliced into a set of N_z depth layers parallel to the hologram plane and located between $z_{\min}$ and $z_{\max}$. We call d_z the distance between each depth layer. N_z is set such that the separation between two consecutive layers remains invisible [31]. The depth layers are numbered from 0 to $N_z - 1$, from the farthest to the nearest to the hologram plane. The depth layers and the hologram are sampled on a regular 2D grid of resolution $(N_x \times N_y)$ with sampling pitch p.

The depth layers are classified into two categories: the wave-field layers and the point-source layers, as shown in Figure 4. This classification is performed depending on the number of scene points within each layer: if the number of scene points M_d within layer d exceeds a threshold value $M_{d,\max}$, this layer is considered to be a wave-field layer, otherwise it is considered to be a point-source layer. The farthest and nearest depth layers are always considered to be wave-field layers.

Each depth layer d operates as a surface source of light which emits a complex wave o_d given by

$$o_d(x, y) = A_d(x, y) \exp\left[j\phi_d(x, y)\right], \tag{1}$$

where $A_d(x, y)$ is the amplitude of the (x, y) point within layer d, calculated using illumination formulas, and $\phi_d(x, y) \in [0; 2\pi[$ is its phase, set to a random value to render a diffusive scene.

Light scattered by the scene is numerically propagated and shielded from one layer to another using the recurrence formula

$$\begin{cases} u_0(x, y) = o_0(x, y) \\ u_d(x, y) = o_d(x, y) + \mathcal{O}_d\left\{\mathcal{P}_{d_z}\left\{u_{d-1}\right\}(x, y)\right\} \quad \text{for } d > 0, \end{cases} \tag{2}$$

where u_d is the total complex wave scattered by layer d, operator $\mathcal{O}_d$ stands for light shielding by layer d, and operator $\mathcal{P}_z$ stands for the numerical propagation of light between two parallel planes separated by a distance z. These numerical operations are performed using either a point-source or a wave-field approach depending on which category layer d belongs to.

Finally, the complex wave scattered by layer $N_z - 1$ is numerically propagated to the hologram plane to obtain the final CGH, according to

$$H(x, y) = \mathcal{P}_{z_{\min}}\left\{u_{N_z-1}\right\}(x, y). \tag{3}$$

3 Light propagation

This section presents the two approaches used by the proposed method for numerical light propagation. In the following, we call $\mathcal{P}_z^w$ and $\mathcal{P}_z^s$ the operators used to compute the propagation of light between two parallel planes separated by a distance z using the wave-field and point-source approaches, respectively.

3.1 Wave-field layers

The propagation of complex waves scattered by wave-field layers is numerically computed using the Angular Spectrum formula [29], which expresses light diffraction between two parallel planes separated by a distance z as

$$\mathcal{P}_z^w\left\{u_d\right\}(x, y) = \mathcal{F}^{-1}\left\{\mathcal{F}\left\{u_d\right\} \mathrm{e}^{j2\pi z\sqrt{\lambda^{-2}-f_x^2-f_y^2}}\right\}(x, y), \tag{4}$$

where λ is the wavelength of light, f_x and f_y are the spatial frequencies, and $\mathcal{F}$ and $\mathcal{F}^{-1}$ are respectively the forward and inverse Fourier Transform. These transforms can be computed using the Fast Fourier Transform algorithm (FFT). The calculation of the right-hand side member of (4) has computational complexity $O(Nlog(N))$, where $N = (N_x \times N_y)$ is the number of hologram pixels.

3.2 Point-source layers

To compute the propagation of complex waves scattered by point-source layers, scene points located within these layers are considered as spherical light sources. Light propagation is therefore computed as the sum of spherical waves scattered by each point. The complex wave scattered by a point source k at coordinates (x_k, y_k, z) is given by the Angular Spectrum [29] as

$$w_k(x, y) = u_d(x_k, y_k)\mathcal{F}^{-1}\left\{\mathrm{e}^{j2\pi z\sqrt{\lambda^{-2}-f_x^2-f_y^2}}\right\} \otimes \delta(x - x_k, y - y_k), \tag{5}$$

where $u_d(x_k, y_k)$ is the complex amplitude of the point and $\otimes$ is the convolution operator.

Since convolving a function with a position-shifted Dirac delta shifts it by the same amount, knowing the inverse Fourier transform term in (5) beforehand allows to compute w_k simply by scaling this term with $u_d(x_k, y_k)$, followed by a shifting operation. To speed up the computation, we use

a pre-calculated LUT, as proposed in [20]. The LUT T is pre-computed as

$$T(x,y,z) = \mathcal{F}^{-1}\left\{ \mathrm{e}^{j2\pi z\sqrt{\lambda^{-2}-f_x^2-f_y^2}} \right\} h(x,y,z), \qquad (6)$$

h being a window function used to restrict the region of contribution of a given point source, equal to one within the region of contribution of the point and zero elsewhere. This function limits the spatial frequencies of the complex wave to avoid aliasing in the CGH.

According to the Nyquist Sampling Theorem, the maximum spatial frequency $f_{\max}$ that can be represented with a sampling pitch p is given by $f_{\max} = (2p)^{-1}$. The grating equation [29] gives the relation between the maximum spatial frequency $f_{\max}$ and the maximum diffraction angle θ as $\sin(\theta) = \lambda f_{\max}$. Therefore, the region of contribution of a point source at depth z is given by its maximum radius $R_{\max}$ by

$$R_{\max}(z) = z\tan(\theta) = z\tan\left(\arcsin\left(\frac{\lambda}{2p}\right)\right), \qquad (7)$$

as shown in Figure 5. The window function h can thus be defined as

$$h(x,y,z) = \begin{cases} 1 & \text{if } \sqrt{x^2+y^2} < R_{\max}(z) \\ 0 & \text{otherwise.} \end{cases} \qquad (8)$$

Thanks to its geometrical symmetry and to the window function h defined by (8), the LUT needs not be computed for every (x,y) values. Instead of it and in order to limit its number of samples, T is pre-computed only within the upper quarter of the square circumscribing the disk defined by h. Therefore, the number of samples $N_{T,z}$ of the LUT for depth z is given by

$$N_{T,z} = \left(\frac{R_{\max}(z)}{p}\right)^2 = \left[\frac{z}{p}\tan\left(\arcsin\left(\frac{\lambda}{2p}\right)\right)\right]^2. \qquad (9)$$

Then, light propagation between two parallel planes separated by a distance z is computed simply by addressing this pre-calculated LUT, such that

$$\mathcal{P}_z^s\left\{u_d\right\}(x,y) = \sum_{k=0}^{M_d-1} u_d(x_k,y_k)T(x-x_k,y-y_k,z), \qquad (10)$$

where M_d is the number of self-luminous points within the source plane. The calculation of the right-hand side member of (10) has computational complexity $O(N_{T,z}M_d)$.

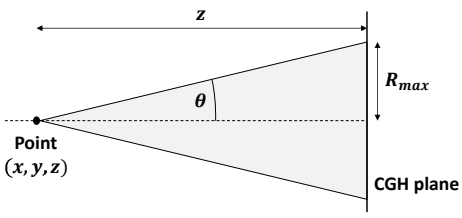

Figure 5: Region of contribution of a given point source.

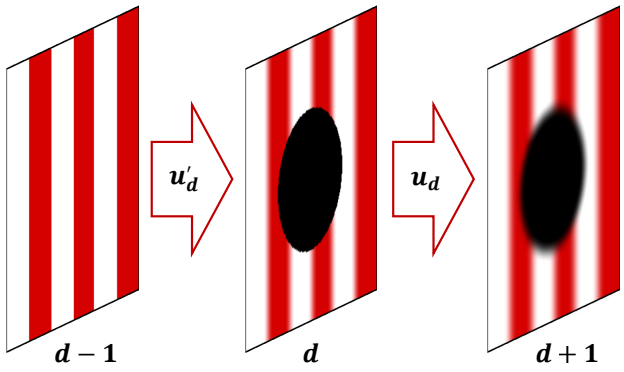

Figure 6: Light shielding using the binary cross-section mask function.

4 Light shielding

In this section, we present how light shielding is performed using the proposed method. We develop our method from the simplest case, in which all the depth layers are wave-field layers, to the most general case, in which several point-source layers are stacked between each two wave-field layers.

4.1 Case 1: Light shielding by wave-field layers

We first consider the case in which all the depth layers contain a sufficient number of scene points to be considered as wave-field layers. In that case, light shielding is performed using a binary cross-section mask function, as proposed in [30].

The procedure is given in Figure 6. An occluding obstacle is located within layer d. The light wave u'_d incident on layer d is given by

$$u'_d(x,y) = \mathcal{P}_{d_z}^w\left\{u_{d-1}\right\}(x,y), \qquad (11)$$

where u_{d-1} is the total complex wave scattered by layer $d-1$.

Part of u'_d is shielded by the occluding scene points and its amplitude vanishes in the area of the obstacle. This is expressed by multiplying u'_d by a binary mask function m_d that has value zero on the obstacle and one elsewhere. If the scene points within layer d also emit a wave o_d, the total complex wave scattered by layer d is therefore given by

$$u_d(x,y) = o_d(x,y) + m_d(x,y)u'_d(x,y). \qquad (12)$$

Light scattered by the scene is therefore numerically propagated and shielded from one layer to another using the recurrence formula

$$\begin{cases} u_0(x,y) = o_0(x,y) \\ u_d(x,y) = o_d(x,y) + m_d(x,y)\mathcal{P}_{d_z}^w\left\{u_{d-1}\right\}(x,y) \quad \forall d > 0 \end{cases} \qquad (13)$$

4.2 Case 2: One point-source layer between each two wave-field layers

Let's now consider the case in which a single point-source layer d is located between two wave-field layers $d-1$ and $d+1$. An occluding obstacle is located within layer d. A straightforward way to compute light shielding by layer d is to use (12). The complex wave incident on layer $d+1$ is therefore given by

$$\begin{aligned} u'_{d+1}(x,y) &= \mathcal{P}^s_{d_z}\left\{u_d\right\}(x,y) \\ u'_{d+1}(x,y) &= \mathcal{P}^s_{d_z}\left\{o_d + m_d \mathcal{P}^w_{d_z}\left\{u_{d-1}\right\}\right\}(x,y). \end{aligned} \tag{14}$$

However, while perfectly adapted to wave-field layers, whose complex wave is numerically propagated using $\mathcal{P}^w_z$, light shielding using the binary mask function is not suited to point-source layers. Indeed, at each diffraction step, the complex wave scattered by the scene spreads gradually from one layer to another, and the wave u'_d incident on layer d may spread out on a large number of samples. Thus, even if layer d contains only a few scene points for which o_d is non-zero, the total complex wave u_d scattered by this layer may have a large number of non-zero values. Each non-zero value of u_d is therefore considered as a spherical light source by $\mathcal{P}^s_z$. Since the computational complexity of $\mathcal{P}^s_z$ is dependent on the number of spherical light sources within the source layer, this light shielding technique is highly inefficient for point-source layers.

Instead, to compute light shielding by point-source layers efficiently, we use a binary aperture function a_d given by

$$a_d(x,y) = 1 - m_d(x,y), \tag{15}$$

as proposed in [37]. By substituting (15) in (14), the complex wave u'_{d+1} incident on layer $d+1$ becomes

$$\begin{aligned} u'_{d+1} &= \mathcal{P}^s_{d_z}\left\{o_d + (1-a_d)\mathcal{P}^w_{d_z}\left\{u_{d-1}\right\}\right\} \\ &= \mathcal{P}^s_{d_z}\left\{\mathcal{P}^w_{d_z}\left\{u_{d-1}\right\} + o_d - a_d\mathcal{P}^w_{d_z}\left\{u_{d-1}\right\}\right\} \\ &= \mathcal{P}^w_{2d_z}\left\{u_{d-1}\right\} + \mathcal{P}^s_{d_z}\left\{o_d - a_d\mathcal{P}^s_{d_z}\left\{u_{d-1}\right\}\right\} \\ &= \mathcal{P}^w_{2d_z}\left\{u_{d-1}\right\} + \mathcal{P}^s_{d_z}\left\{\hat{u}_d\right\}, \end{aligned} \tag{16}$$

since

$$\mathcal{P}^s_{d_z} \equiv \mathcal{P}^w_{d_z} \quad \text{and} \quad \mathcal{P}^s_{d_z}\left\{\mathcal{P}^w_{d_z}\left\{u_{d-1}\right\}\right\} \equiv \mathcal{P}^w_{2d_z}\left\{u_{d-1}\right\}, \tag{17}$$

according to definitions of $\mathcal{P}^w_z$ and $\mathcal{P}^s_z$.

The calculation of u'_{d+1} using the binary aperture involves computing one more propagation than using the binary mask. However, $\mathcal{P}^s_{d_z}\left\{u_{d-1}\right\}$ needs to be calculated only within the region defined by aperture a_d, which corresponds to the coordinates of scene points located within layer d. Therefore, the number of non-zero values of $\hat{u}_d$ remains equal to the number of scene points for which o_d is non-zero. Using the binary aperture function has the great advantage of not increasing the number of spherical light sources for which the complex wave has to be computed at each diffraction step. This technique is therefore highly efficient for light shielding by point-source layers.

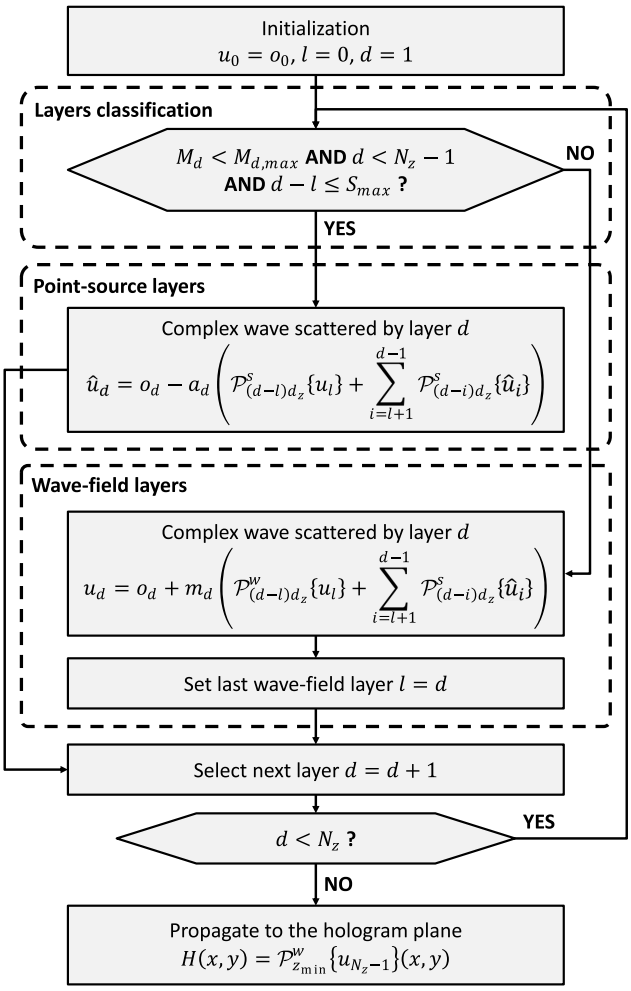

Figure 7: Block-diagram of the proposed method.

Light scattered by the scene is therefore numerically propagated and shielded from one layer to another, using the recurrence formula

$$\begin{cases} u_0 = o_0 & \\ \hat{u}_d = o_d - a_d\mathcal{P}^s_{d_z}\left\{u_{d-1}\right\} & \forall d > 0, d \in S \\ u_d = o_d + m_d\left(\mathcal{P}^w_{2d_z}\left\{u_{d-2}\right\} + \mathcal{P}^s_{d_z}\left\{u_{d-1}\right\}\right) & \forall d > 1, d \in W \end{cases} \tag{18}$$

where S and W are the sets of point-source and wave-field layers, respectively.

4.3 Case 3: Several point-source layers between each two wave-field layers

We now consider the most general case, in which several point-source layers are stacked between each two wave-field layers. The overall block-diagram of the method is shown in Figure 7.

To compute light scattered by the scene, (18) is generalized

$$
\begin{aligned}
u_{d+1} &= o_{d+1} + m_{d+1}\mathcal{P}^w_{d_z}\{u_d\} \\
&= o_{d+1} + m_{d+1}\mathcal{P}^w_{d_z}\left\{o_d + (1-a_d)\left(\mathcal{P}^w_{(d-l)d_z}\{u_l\} + \sum_{i=l+1}^{d-1}\mathcal{P}^s_{(d-i)d_z}\{\hat{u}_i\}\right)\right\} \\
&= o_{d+1} + m_{d+1}\left(\mathcal{P}^s_{d_z}\{\hat{u}_d\} + \mathcal{P}^w_{(d+1-l)d_z}\{u_l\} + \sum_{i=l+1}^{d-1}\mathcal{P}^s_{(d+1-i)d_z}\{\hat{u}_i\}\right) \\
&= o_{d+1} + m_{d+1}\left(\mathcal{P}^w_{(d+1-l)d_z}\{u_l\} + \sum_{i=l+1}^{d}\mathcal{P}^s_{(d+1-i)d_z}\{\hat{u}_i\}\right), \qquad (19)
\end{aligned}
$$

where

$$
\begin{aligned}
\hat{u}_{d+1} &= o_{d+1} - a_{d+1}\mathcal{P}^s_{d_z}\{u_d\} \\
&= o_{d+1} - a_{d+1}\left(\mathcal{P}^s_{(d+1-l)d_z}\{u_l\} + \sum_{i=l+1}^{d}\mathcal{P}^s_{(d+1-i)d_z}\{\hat{u}_i\}\right). \qquad (20)
\end{aligned}
$$

to

$$
\begin{cases}
u_0 = o_0 \\
\hat{u}_d = o_d - a_d\left(\mathcal{P}^s_{(d-l)d_z}\{u_l\} + \sum_{i=l+1}^{d-1}\mathcal{P}^s_{(d-i)d_z}\{\hat{u}_i\}\right) \\
\qquad\qquad \forall d, 0 \leq l < d, d \in S \\
u_d = o_d + m_d\left(\mathcal{P}^w_{(d-l)d_z}\{u_l\} + \sum_{i=l+1}^{d-1}\mathcal{P}^s_{(d-i)d_z}\{\hat{u}_i\}\right) \\
\qquad\qquad \forall d, 0 \leq l < d, d \in W
\end{cases} \qquad (21)
$$

where l is the last wave-field layer. Since the binary aperture a_d has value one only at the coordinates of scene points located within layer d, $\hat{u}_d$ needs to be calculated within this region only. The computational complexity of this calculation is thus $O(M_d M + M_d N_{T,(d-l)d_z})$, where M is the total number of scene points within the depth layers located between d and the last wave-field layer l: $M = \sum_{i=l+1}^{d-1} M_i$.

Let $k = d - l - 1$ be the number of point-source layers between the last wave-field layer l and the current layer d. We call $P(n)$ the property which states that (21) and (13) give the same complex field u_d for all $k \leq n$. We demonstrate that $P(n)$ holds for all $n \geq 0$. For $n = 0$, (21) becomes

$$
u_d = o_d + m_d\mathcal{P}^w_{d_z}\{u_{d-1}\} \quad \forall d > 0, \qquad (22)
$$

which we recognize to be equal to u_d given by (13). $P(n)$ therefore holds for $n = 0$.

We now assume that $P(n)$ holds. For $k = n + 1$, u_{d+1} is therefore given by (19). This equation shows that if $P(n)$ holds, $P(n+1)$ also holds. As a consequence, $P(n)$ holds for all $n \geq 0$. Since we use the same succession of layers as defined in the formulation of $P(n)$, our method gives the exact same results as those given by (13).

5 Layers classification

The first step to implement the proposed method is to determine the value of $M_{d,\max}$. We call T^w_d the time needed to compute light propagation and light shielding for layer d using the wave-field approach, and T^s_d the time needed to compute it using the point-source approach.

T^w_d is only dependent on the number of hologram pixels N and is given by

$$
T^w_d = \alpha N log(N) + \beta N, \qquad (23)
$$

where α and β are constant coefficients.

On the other hand, T^s_d is given by the sum of the time needed to compute the propagation of complex wave scattered by layer d using $\mathcal{P}^s$ and the time needed to compute light shielding by layer d using aperture function a_d. It is therefore given by

$$
T^s_d = \gamma\left(M_d N_{T,z_d} + M_d M + M_d N_{T,(d-l)d_z}\right), \qquad (24)
$$

where γ is a constant coefficient and M is the total number of scene points within the depth layers located between d and the last wave-field layer l. The values of α, β and γ are dependent on the implementation and on the computing system used.

To maximize the efficiency of the proposed method, $M_{d,\max}$ must be set such that

$$
T^w_d = T^s_d \Leftrightarrow M_{d,\max} = \frac{\alpha N log(N) + \beta N}{\gamma(N_{T,z_d} + M + N_{T,(d-l)d_z})}. \qquad (25)
$$

To find the numerical values of the coefficients α, β and γ, we measured the calculation time of the point-source and wave-field approaches for one million randomly generated depth layers with different numbers of scene points. We then found these values using the Gnuplot implementation of the nonlinear least-squares Levenberg-Marquardt algorithm [38].

It must be noted that the calculation of (21) requires the complex waves $\hat{u}_i$ scattered by depth layers $i \in \{l+1, ..., d-1\}$ to be kept in memory. Since this may require a huge amount

of memory, we restrict the number of consecutive point-source layers to be less or equal to a maximum value $S_{\max}$. The value of $S_{\max}$ may be adjusted depending on the maximum memory amount of the computing system used.

6 Graphics Processing Unit implementation

6.1 CUDA thread organization and memory model

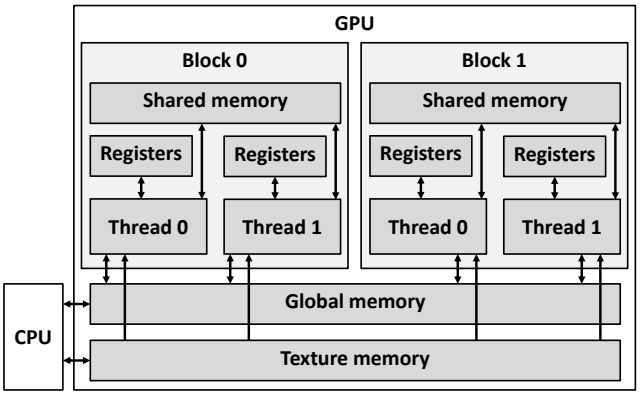

Figure 8: CUDA thread organization and memory model

The proposed method was implemented in C++/CUDA on a PC system employing an Intel Core i7-4930K CPU operating at 3.40 GHz, a main memory of 16 GB, three GPUs NVIDIA GeForce GTX 780Ti, and an operating system of Microsoft Windows 8. To compute the CGH patterns for the three colors simultaneously, we used one CPU thread and one GPU per color. In the implementation, all the CGH computation is done by the GPUs. The CPU threads are only used to load the input 3D scene, launch CUDA kernels and save the computed CGH into an output file. Finally, to achieve best performance on the GPU, all the computations are performed using single precision.

Figure 8 shows the CUDA thread organization and memory model used by the NVIDIA GeForce GTX 780Ti. The parallel portions of an application to be executed on the GPU are called kernels. Each kernel is executed in parallel by many threads, which are organized in blocs. Each thread can independently process and store data using the on-chip registers. Threads within a block can cooperate and exchange data through the on-chip shared memory, which has a short-latency but limited capacity. The CPU and GPU threads can exchange data through the global memory, which has a large capacity but long latency and limited bandwidth. Additionally, the GPU threads can also access to a read-only texture memory, which is cached on-chip. The texture cache is optimized for 2D spatial locality, so threads that read texture addresses that are close together will achieve best performance.

Algorithm 1 Kernel1 pseudo-code

Require: $\mathcal{F}\{u_d\}$
Require: z
Ensure: U_d
1: **for all** (f_x, f_y) in parallel **do**
2: $\quad U_d(f_x, f_y) \leftarrow \mathcal{F}\{u_d\}(f_x, f_y) \exp\left(j2\pi z\sqrt{\lambda^{-2} - f_x^2 - f_y^2}\right)$
3: **end for**

6.2 Implementation of $\mathcal{P}_z^w$

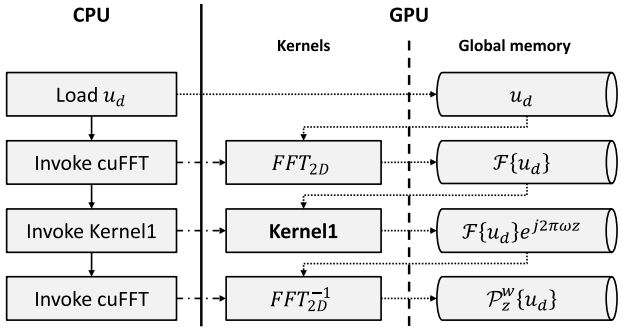

Figure 9: Software structure for the computation of complex waves scattered by wave-field layers

Figure 9 shows the software structure for the computation of $\mathcal{P}_z^w$. The CPU first loads the complex wave u_d scattered by the source layer d into the GPU global memory. Then, the Fourier transform in (4) is performed on the GPU using the CUDA cuFFT library by NVIDIA. This library uses the Cooley-Tukey algorithm [39] to optimize the performance of any transform size that can be factored as $2^a 3^b 5^c 7^d$, where a, b, c and d are non-negative integers.

Once the Fourier transform is performed, the CPU invokes kernel *Kernel1*, whose pseudo-code is given in Algorithm 1. *Kernel1* multiplies every sample of $\mathcal{F}\{u_d\}$ with the transfer function of (4) in parallel, with one thread per sample. Finally, once all the wave-field layers have been processed, the inverse Fourier transform in (4) is performed on the GPU using the cuFFT library.

6.3 Implementation of $\mathcal{P}_z^s$

Figure 10 shows the software structure for the computation of $\mathcal{P}_z^s$. The CPU first loads the complex wave u_d scattered by the source layer and the coordinates $\{(x_k, y_k)\}_{0 \le k < M_d}$ of scene points located within the layer into the GPU global memory. Then, the CPU invokes kernel *Kernel2*, whose pseudo-code is given in Algorithm 2.

Kernel2 computes the numerical propagation $\mathcal{P}_z^s$ on every sample of the output plane in parallel, with one thread per sample. Since the computation of $\mathcal{P}_z^s$ involves one calculation per scene point within the source layer per sample of the output plane, all the threads executing *Kernel2* must access u_d and $\{(x_k, y_k)\}_{0 \le k < M_d}$ data simultaneously. However, this data is stored in the global memory, which has a long latency.

Algorithm 2 Kernel2 pseudo-code

Require: u_d
Require: z
Require: $\{(x_k, y_k)\}_{0 \le k < M_d}$
Ensure: $\mathcal{P}_z^s \{u_d\}$
for $k \in \{0, .., M_d - 1\}$ in parallel **do**
  $x_k^s \leftarrow x_k$ ▷ Load x_k in shared memory
  $y_k^s \leftarrow y_k$ ▷ Load y_k in shared memory
  $u_k^s \leftarrow u_d(x_k, y_k)$ ▷ Load $u_d(x_k, y_k)$ in shared memory
end for
Synchronize threads
for all (x, y) in parallel **do**
  $r \leftarrow 0$ ▷ Initialize register value
  for $k \in \{0, .., M_d - 1\}$ **do**
    $X \leftarrow |x - x_k^s|$
    $Y \leftarrow |y - y_k^s|$
    if $X < R_{\max}(z)$ AND $Y < R_{\max}(z)$ **then**
      $r \leftarrow r + u_k^s T(X, Y, z)$
    end if
  end for
  $\mathcal{P}_z^s \{u_d\}(x, y) \leftarrow r$ ▷ Store output in global memory
end for

Therefore, to reduce and optimize data transfers, *Kernel2* starts out by loading $\{(x_k, y_k, u_d(x_k, y_k))\}_{0 \le k < M_d}$ into the shared memory, which has a much shorter latency than global memory. Once this is done, threads must be synchronized to wait until all the data has been properly loaded into shared memory.

Kernel2 then fetches T to compute the spherical waves scattered by each point within the source layer. Since all the threads compute the spherical wave scattered by point k for different samples of the output plane simultaneously, consecutive threads are expected to fetch adjacent samples of T. Since the texture memory is optimized for 2D spatial locality memory accesses, T is stored in this memory to achieve better performance.

Each sample of the LUT is stored as an 8 bytes single precision complex value in the texture memory. The total amount of memory needed to store T is therefore given by

$$N_T = 8 S_{\max} N_{T, S_{\max} d_z}. \tag{26}$$

For $S_{\max} = 100$, $N_z = 2048$, $\lambda = 640$nm, $p = 4.8\mu$m and $z_{\max} = 10$cm, the total amount of memory needed to store T is $N_T = 3.7$MB. Most current desktop GPUs have more than 512MB of available memory, so this method does not require the use of a professional GPU.

Kernel2 therefore fetches the LUT and performs a multiply-accumulate operation for each scene point k. To limit the number of accesses to global memory, this multiply-accumulate operation is performed using an on-chip register, and the final result is stored only once into the global memory.

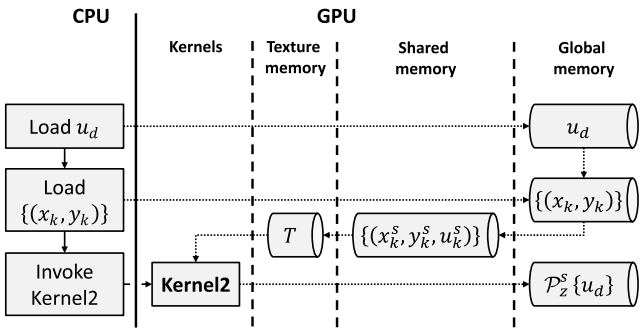

Figure 10: Software structure for the computation of complex waves scattered by point-source layers

7 Experimental results

Table 1 shows the hologram parameters used for the experiments. The hologram is sampled on a regular 2D grid of resolution (7680×4320) with sampling pitch $p = 4.8\mu$m. The wavelengths are set to 640nm, 532nm and 473nm for the Red, Green and Blue channels, respectively. Finally, $S_{\max}$ is set to 100.

We compared the proposed method with GPU implementations of two other methods: (1) the point-source method without occlusion effect proposed in [20], which computes complex wave scattered by each layer using a point-source approach, and (2) the wave-field method with occlusion effect proposed in [30], which computes complex wave scattered by each layer using a wave-field approach. We adapted both methods to produce colorful complex modulation CGH.

7.1 Input 3D scenes

For the experiments, we used three different test scenes, whose CG images are shown in Figure 11. The scenes are sliced into a set of N_z depth layers parallel to the hologram plane and located between $z_{\min} = 0$ and $z_{\max} = 4$cm. N_z is set to 512, 1024 and 2048 for *Dice1*, *Dice2* and *City*, respectively. Figure 15 shows the number of scene points per layer for the three input scenes. As shown in this figure, all the depth layers contain scene points, except those located between the background and the first dice in *Dice1* and *Dice2*.

Parameter	Value
Hologram resolution	(7680×4320)
Pixel pitches	$(4.8\mu\text{m} \times 4.8\mu\text{m})$
Red wavelength	640nm
Green wavelength	532nm
Blue wavelength	473nm

Table 1: Hologram parameters used for the experiments

(a) *Dice1* (b) *Dice2* (c) *City*

Figure 11: Computer graphics images of the three test scenes used for the experiments.

Figure 12: Numerical reconstructions of the CGHs generated by the point-source method (first row), the wave-field method (second row), and the proposed method (third row).

(a) Focused on the blue dice (b) Focused on the red dice (c) Focused on the green dice

Figure 13: Optical reconstructions of scene *Dice2* focused on different objects. These pictures were taken with the NICTs 8K4K holographic display.

(a) Left view (b) Center view (c) Right view

Figure 14: Optical reconstructions of scene *Dice2* captured from different viewpoints. These pictures were taken with the NICTs 8K4K holographic display.

Scene	**Scene points**	**Scene points considered as point sources**	**Method**	**Calculation time**		
				Total	**Per layer**	**Per point**
Dice1	12,820,048	3,418,169 (26.7%)	Wave-field	21.79s (100%)	42.56ms	1.70μs
			Point-source	23.45s (108%)	45.80ms	1.83μs
			Proposed	12.22s (56.1%)	23.87ms	0.95μs
Dice2	16,736,640	6,716,234 (40.1%)	Wave-field	42.22s (100%)	41.23ms	2.52μs
			Point-source	30.64s (72.6%)	29.92ms	1.83μs
			Proposed	21.79s (51.6%)	21.28ms	1.30μs
City	47,648,608	13,718,096 (28.8%)	Wave-field	85.34s (100%)	41.67ms	1.79μs
			Point-source	86.99s (102%)	42.48ms	1.83μs
			Proposed	69.76s (81.7%)	34.06ms	1.46μs

Table 2: CGH computation times for the three test scenes using the wave-field, point-source and proposed methods.

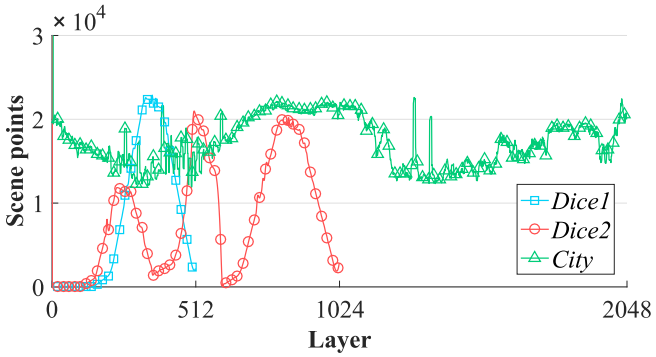

Figure 15: Number of scene points per layer for the three scenes.

7.2 Numerical and optical reconstructions

Figure 12 shows the scene images numerically reconstructed from the CGH patterns of the three test scenes generated by the point-source method (first row), the wave-field method (second row), and the proposed method (third row). As shown in the first row, the point-source method does not take into account occlusions in the scene, and the objects appear therefore semi-transparent, strongly limiting the realism of the displayed image. On the other hand, occlusions between objects in the scene are accurately reproduced using the wave-field and proposed methods without producing any visible artifact (rows 2 and 3). These results confirm the need for CGH methods to properly take into account occlusions between objects in a scene to provide a realistic three-dimensional illusion.

To experimentally verify these numerical results, we displayed the generated hologram of scene *Dice2* on the National Institute of Information and Communications Technology's (NICT) 8K4K holographic display [40]. Figure 13 shows the optically reconstructed scene images recorded with a camera focusing on different depths. When the camera is focused on a given object, the rest of the scene is blurred, and vice versa. This demonstrates that the proposed method provides an accurate accommodation cue of the 3D scene. Figure 14 shows the optical reconstructions captured from three different viewpoints. As seen in this figure, the proposed method is able to produce motion parallax with correct occlusion effect.

7.3 Analysis of the CGH calculation time

Table 2 shows the CGH computation times for the three test scenes using the wave-field approach, the point-source approach and the proposed method. *Dice1* is sliced into 512 depth layers and is sampled by a total number of $12,820,048$ points, *Dice2* is sliced into 1024 depth layers and is sampled by $16,736,640$ points, and *City* is sliced into 2048 depth layers and is sampled by $47,648,608$ points.

As shown in Table 2, using the wave-field method, the average CGH calculation time per layer remains constant for the three test scenes. This means that the total calculation time increases linearly with the number of layers and is not dependent on the total number of scene points. On the contrary, using the point-source method, the average CGH calculation time for one scene point remains constant for the three test scenes. This means that the total calculation time increases linearly with the number of scene points and is not dependent on the number of depth layers.

As a result, the point-source method is less efficient than the wave-field method when the number of points per layer is high, as in scenes *Dice1* and *City*, and is more efficient when the average number of points within each layer is low, as in scene *Dice2*. Using the point-source method, the total CGH calculation time has been increased by 8% and 2% for *Dice1* and *City*, and decreased by 27.4% for *Dice2*, compared to the wave-field method. It must be noted that the point-source approach that we implemented here does not take into account occlusions between objects in the scene. A light shielding operation, performed as a visibility test to check the existence of obstacles between an object point and each sampling point in the hologram plane, would have for effect to increase by a constant amount the average CGH calculation time per scene point.

As shown in Table 2, the complex waves scattered by 26.7%, 40.1% and 28.8% of scene points were computed by the proposed method using the point-source approach for *Dice1*, *Dice2* and *City*, respectively. As a consequence, the total calculation time using the proposed method is dependent both on the number of layers and on the number of scene points. By combining these two approaches, the proposed method takes advantages from both of them and is therefore always more efficient. Using the proposed method, the total CGH calculation time has been decreased by 43.9%, 48.4% and 18.3% for *Dice1*, *Dice2* and *City*, respectively, compared to the wave-field method, and by 47.9%, 28.9% and 19.8% for *Dice1*, *Dice2* and *City*, respectively, compared to the point-source method. These experimental results confirm the performance superiority of the proposed method over the point-source and wave-field methods in terms of computation time.

8 Conclusion

In this paper, we proposed a fast Computer Generated Hologram computation method with occlusion effect based on a hybrid point-source/wave-field approach. Our algorithm consists of three steps. First, the 3D scene is sliced into several depth layers parallel to the hologram plane. Then, light scattered by the scene is propagated and shielded from one layer to another, starting from the farthest layer. For each depth layer, light propagation and light shielding are performed using either a point-source or a wave-field approach according to a threshold criterion on the number of points within the layer. Finally, we compute light propagation from the nearest layer to the hologram plane to obtain the final CGH.

Experimental results reveal that the proposed method accurately takes into account occlusions between objects in the

scene and is able to produce a correct accommodation cue. Furthermore, the CGH calculation time has been reduced up to 47.9% and 48.4% compared to the point-source and wave-field approaches, respectively. This confirms the performance superiority of the proposed method over the point-source and wave-field approaches in terms of computation time.

9 Acknowledgment

This work has been achieved within the Institute of Research and Technology b<>com, dedicated to digital technologies. It has been funded by the French government through the National Research Agency (ANR) Investment referenced ANR-A0-AIRT-07.

The authors would like to thank Kenji Yamamoto's team from the NICT for their warm welcome, for the very interesting discussions and for giving us access to their holographic display.

References

[1] Charles Wheatstone. Contributions to the Physiology of Vision.–Part the First. On Some Remarkable, and Hitherto Unobserved, Phenomena of Binocular Vision. *Philosophical Transactions of the Royal Society of London*, 128, January 1838.

[2] Neil A. Dodgson. Analysis of the viewing zone of multi-view autostereoscopic displays. volume 4660, 2002.

[3] David M. Hoffman, Ahna R. Girshick, Kurt Akeley, and Martin S. Banks. Vergence–accommodation conflicts hinder visual performance and cause visual fatigue. *Journal of Vision*, 8(3):33, March 2008.

[4] Ulf Schnars and Werner Jüptner. *Digital Holography: Digital Hologram Recording, Numerical Reconstruction, and Related Techniques*. Springer Science & Business Media, December 2005.

[5] B. R. Brown and A. W. Lohmann. Complex Spatial Filtering with Binary Masks. *Applied Optics*, 5(6), June 1966.

[6] John S. Underkoffler. Occlusion processing and smooth surface shading for fully computed synthetic holography. In *Practical Holography XI and Holographic Materials III*, volume Proc. SPIE 3011, April 1997.

[7] Rick H-Y Chen and Timothy D Wilkinson. Computer generated hologram with geometric occlusion using GPU-accelerated depth buffer rasterization for three-dimensional display. *Applied optics*, 48(21):4246–4255, July 2009.

[8] Hao Zhang, Neil Collings, Jing Chen, Bill Crossland, Daping Chu, and Jinghui Xie. Full parallax three-dimensional display with occlusion effect using computer generated hologram. *Optical Engineering*, 50(7), 2011.

[9] Hao Zhang, Qiaofeng Tan, and Guofan Jin. Full parallax three-dimensional computer generated hologram with occlusion effect using ray casting technique. *Journal of Physics: Conference Series*, 415(1), February 2013.

[10] J. L. Juárez-Pérez, A. Olivares-Pérez, and L. R. Berriel-Valdos. Nonredundant calculations for creating digital Fresnel holograms. *Applied Optics*, 36(29):7437–7443, October 1997.

[11] Hiroshi Yoshikawa, Susumu Iwase, and Tadashi Oneda. Fast computation of Fresnel holograms employing difference. In *Practical Holography XIV and Holographic Materials VI*, volume 3956, May 2000.

[12] Kyoji Matsushima and Masahiro Takai. Recurrence Formulas for Fast Creation of Synthetic Three-Dimensional Holograms. *Applied Optics*, 39(35), December 2000.

[13] Takeshi Yamaguchi, Gen Okabe, and Hiroshi Yoshikawa. Real-time image plane full-color and full-parallax holographic video display system. *Optical Engineering*, 46(12), 2007.

[14] Hiroshi Yoshikawa, Takeshi Yamaguchi, and Ryo Kitayama. Real-Time Generation of Full Color Image Hologram with Compact Distance Look-up Table. In *Advances in Imaging*, OSA Technical Digest (CD). Optical Society of America, April 2009.

[15] Tomoyoshi Shimobaba, Nobuyuki Masuda, and Tomoyoshi Ito. Simple and fast calculation algorithm for computer-generated hologram with wavefront recording plane. *Optics Letters*, 34(20):3133–3135, October 2009.

[16] Tomoyoshi Shimobaba, Hirotaka Nakayama, Nobuyuki Masuda, and Tomoyoshi Ito. Rapid calculation algorithm of Fresnel computer-generated-hologram using look-up table and wavefront-recording plane methods for three-dimensional display. *Optics Express*, 18(19), September 2010.

[17] Jiantong Weng, Tomoyoshi Shimobaba, Naohisa Okada, Hirotaka Nakayama, Minoru Oikawa, Nobuyuki Masuda, and Tomoyoshi Ito. Generation of real-time large computer generated hologram using wavefront recording method. *Optics Express*, 20(4), February 2012.

[18] Athanasia Symeonidou, David Blinder, Adrian Munteanu, and Peter Schelkens. Computer-generated holograms by multiple wavefront recording plane method with occlusion culling. *Optics Express*, 23(17):22149, August 2015.

[19] Mark E. Lucente. Interactive computation of holograms using a look-up table. *Journal of Electronic Imaging*, 2(1), January 1993.

[20] Seung-Cheol Kim and Eun-Soo Kim. Effective generation of digital holograms of three-dimensional objects using a novel look-up table method. *Applied Optics*, 47(19):D55–D62, July 2008.

[21] Seung-Cheol Kim and Eun-Soo Kim. Fast computation of hologram patterns of a 3d object using run-length encoding and novel look-up table methods. *Applied Optics*, 48(6), February 2009.

[22] Seung-Cheol Kim, Woo-Young Choe, and Eun-Soo Kim. Accelerated computation of hologram patterns by use of interline redundancy of 3-D object images. *Optical Engineering*, 50(9), 2011.

[23] Rick H.-Y. Chen and Timothy D. Wilkinson. Computer generated hologram from point cloud using graphics processor. *Applied Optics*, 48(36), December 2009.

[24] Ivo Hanák, Martin Janda, and Václav Skala. Detail-driven digital hologram generation. *The Visual Computer*, 26(2), February 2010.

[25] Ivo Hanák, Adam Herout, and Pavel Zemčík. Acceleration of detail driven method for hologram generation. *Optical Engineering*, 49(8):085802–085802–9, 2010.

[26] Lukas Ahrenberg, Philip Benzie, Marcus Magnor, and John Watson. Computer generated holography using parallel commodity graphics hardware. *Optics express*, 14(17), August 2006.

[27] Nobuyuki Masuda, Tomoyoshi Ito, Takashi Tanaka, Atsushi Shiraki, and Takashige Sugie. Computer generated holography using a graphics processing unit. *Optics Express*, 14(2), January 2006.

[28] Yasuyuki Ichihashi, Hirotaka Nakayama, Tomoyoshi Ito, Nobuyuki Masuda, Tomoyoshi Shimobaba, Atsushi Shiraki, and Takashige Sugie. HORN-6 special-purpose clustered computing system for electroholography. *Optics Express*, 17(16), August 2009.

[29] Joseph W. Goodman. *Introduction to Fourier Optics*. Roberts and Company Publishers, Englewood, Colo, 3rd edition, 2005.

[30] A. W. Lohmann. Three-dimensional properties of wave-fields. *Optik*, 51, 1978.

[31] Muharrem Bayraktar and Meriç Özcan. Method to calculate the far field of three-dimensional objects for computer-generated holography. *Applied Optics*, 49(24), August 2010.

[32] Kyoji Matsushima and Akinobu Kondoh. A wave-optical algorithm for hidden-surface removal in digitally synthetic full-parallax holograms for three-dimensional objects. In *Practical Holography XVIII: Materials and Applications*, volume Proc. SPIE 5290, June 2004.

[33] Kyoji Matsushima, Hagen Schimmel, and Frank Wyrowski. Fast calculation method for optical diffraction on tilted planes by use of the angular spectrum of plane waves. *Journal of the Optical Society of America A*, 20(9), September 2003.

[34] Kyoji Matsushima and Sumio Nakahara. Extremely high-definition full-parallax computer-generated hologram created by the polygon-based method. *Applied Optics*, 48(34):H54–H63, December 2009.

[35] Tomoyoshi Shimobaba, Takashi Kakue, and Tomoyoshi Ito. Acceleration of color computer-generated hologram from three-dimensional scenes with texture and depth information. In *Three-Dimensional Imaging, Visualization, and Display 2014*, volume 9117, pages 91170B–91170B–8, 2014.

[36] Antonin Gilles, Patrick Gioia, Rémi Cozot, and Luce Morin. Complex Modulation Computer-Generated Hologram with Occlusion Effect by a Fast Hybrid Point-source/Wave-field Approach. In *Proceedings of Pacific Graphics 2015*, Beijing, China, July 2015.

[37] Kyoji Matsushima, Masaki Nakamura, and Sumio Nakahara. Silhouette method for hidden surface removal in computer holography and its acceleration using the switch-back technique. *Optics Express*, 22(20):24450–24465, October 2014.

[38] D. Marquardt. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2), June 1963.

[39] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90), 1965.

[40] Takanori Senoh, Yasuyuki Ichihashi, Ryutaro Oi, Hisayuki Sasaki, and Kenji Yamamoto. Study of a holographic TV system based on multi-view images and depth maps. volume 8644, pages 86440A–86440A–15, 2013.